Robotics – Exercise 4 – Blue Team Report

Shlomi Ben-Shushan ID: 311408264
Yiftach Neuman ID: 208305359

Summery

In this report we will explain our implementation details of our algorithm that lets a Krembot forage in an unknown arena without using its current position. We will use this algorithm in the Adversarial Foraging tournament and compete with other classmates. Our team number is 1.

Algorithm Description

General Strategy:

Start scanning for food in spiral movement until find any food, and then search for the nest to drop it there, or until bumped into something, and then start wandering. If not holding food and a teammate is ahead, make a soft-turn, i.e., wheels spins in the same direction but in different speed, in order to spread forager in the arena and make way to food-carrying robots. If bumped into something, make a hard-turn, i.e., wheels spins in opposite directions. Moreover, every random time (1.5-3 minutes), all robots activate spiraling mode similar to the beginning. Note that it is important not to move in spirals after food drop because this way the same positions will be scanned over and over by multiple robots.

Forager's State: the following C++ enum defines the six states we used.

```
enum State { spiralMove, spiralTurn, move, rtb, softTurn, hardTurn };
```

The behavior of the forager in each state will be described in Action section.

Sense: in this section we will describe our usage in the robot's sensors.

1. **RGBA Cameras:** the forager uses the 5 RGBA cameras (three in its front, and two on its sides) tow sense identify colors and measure distances.
2. **Bumpers:** the forager uses the three bumpers in its from to sense collisions.

Interpretation: in this section we will describe the interpretation of the sensing data.

1. **Teammate Ahead:** when one of the three front RGBA cameras senses the team's color.
2. **Opponent Ahead:** when one of the three front RGBA cameras senses the opponent team's color.
3. **Robot Ahead:** when 1 or 2 takes place.
4. **Obstacle Ahead:** when frontal distance is less then 10, and there is no robot ahead.
5. **Nest Ahead:** when the front RGBA camera senses team base's color.
6. **Nest to the Right:** when the right or the right-front RGBA cameras senses team base's color, and the nest is not ahead.
7. **Nest to the Left:** when the left or the left-front RGBA cameras senses team base's color, and the nest is not ahead.
8. **Bumped into something:** when one of the front bumpers is pressed.

Action: In this section we will describe each forager state we have defined, and the behavior of the robot (the actions it will take) in each of the state.

1. **Move:** drive straight until:
   a. Found food → Switch to **RTB** state.
   b. Bumped into something → Evaluate which side is better to turn to and **hard**-turn.
   c. Teammate ahead → Evaluate which side is better to turn to and **soft**-turn.
   d. Obstacle ahead → Evaluate which side is better to turn to and **hard**-turn.
   Note that the evaluations in c. and d. are not the same.
2. **RTB:** drive straight until:
   a. Food dropped → Switch to **Move** state.
   b. Nest (or base) nearby → Drive towards it.
   c. Bumped into something → Evaluate which side is better to turn to and **hard**-turn.
   d. Obstacle ahead → Evaluate which side is better to turn to and **hard**-turn.
   Note that the evaluations in c. and d. are not the same.
3. **Spiral-Move:** drive straight until:
   a. Found food → Switch to **RTB** state.
   b. Bumped into something → Switch to **Move** (and Move state will handle it).
   c. Spiral Timer is finished → Make a 90 degree turn by switching to **Spiral-Turn** state.
4. **Soft-Turn:** drive with linear and angular speed until:
   a. Found food → Switch to **RTB** state.
   b. Turning timer is finished → Switch to **Move**.
5. **Hard-Turn:** drive with angular speed only until:
   a. Turning timer → Switch to **RTB** state if holds food, and to **Move** otherwise.
6. **Spiral-Turn:** drive with angular speed only until:
   a. Turning timer → Calculate the time for reaching the required distance to make a (squared) spiraling movement and switch to **Spiral-Move** state.

Setup() Pseudo-Code:

1. Register the robot for the group with `writeTeamColor()` function.
2. Define team, base, and opponent colors, w.r.t `foragingMsg.outColor` value.
3. Set initial state to Spiral-Move.
4. Set other required numerical attributes.
5. Start sand-timers.

Loop() Pseudo-Code:

1. Read sensors.
2. Interpret sensing data.
3. if it is time to switch to Spiraling-Mode, do it and restart timer.
4. Behave according to the current state, as described above.